DiPLe: Learning Directed Collaboration Graphs for Peer-to-Peer Personalized Learning

Xue Zheng
Electrical and Computer Engineering
The Ohio State University
zheng.1822@osu.edu

Parinaz Naghizadeh
Integrated Systems Engineering
The Ohio State University
naghizadeh.1@osu.edu

Aylin Yener
Electrical and Computer Engineering
The Ohio State University
yener@ece.osu.edu

***Abstract*—We study fully decentralized learning in which agents learn collaborative, yet personalized prediction models. Specifically, when learners' local datasets are non-IID, a collaboratively trained *global* model (such as those learned through most federated learning algorithms to minimize the sum of losses across all agents) may sacrifice the *local* performance on agents' private datasets. To address this issue and enable personalized learning, we propose DiPLe: an algorithm for Directed Personalized Learning. Through our algorithm, each agent identifies "relevant" agents with whom to exchange model information. This leads to a *weighted and directed* collaboration graph. Agents repeatedly update this graph, and then exchange information with neighboring agents on this learned graph, to collaboratively train their personalized models. We provide analytical results on the generalization error bounds and convergence of our proposed learning method. We verify the performance of DiPLe through numerical experiments, and show its advantages in terms of personalization compared to a number of existing federated learning and personalized learning algorithms.**

***Index Terms*—Personalized learning, Decentralized learning, Federated learning.**

I. Introduction

In recent years, there has been increasing interest on Federated Learning (FL) as an architecture for enabling distributed and collaborative learning [1]. This architecture consists of a central server and a number of distributed agents/clients. The central server repeatedly collects locally trained prediction models from (a subset of) these distributed agents, combines them together (e.g. by averaging) to obtain a global model, and communicates this shared model back to the agents. FL has the ability to train a single global model from decentralized data, while respecting the privacy of agents' local datasets.

Despite its many desirable properties, FL suffers from a number of drawbacks, one of which is the lack of *personaliziation* in the learned model. To illustrate this, consider the virtual keyboard, one of the commonly proposed applications of FL, which trains a shared global model to automatically give word suggestions to users. A single shared model may not provide equally accurate recommendations to all users; for instance, an emoji or symbol may have almost opposite meanings to users with different backgrounds. With a shared global model, many users and particularly minority groups, will inevitably experience poor (local) performance. More generally, in many real-world applications, data distributions tend to be highly heterogeneous among different agents, and an FL algorithm learning a global consensus model will be sensitive to the heterogeneity of data distributions [1]. This has led to an interest in developing collaborative learning algorithms that combine the best of both worlds: improving learning rate and generalization by leveraging multiple learners' distributed computation power and datasets, while maintaining the specialized and personalized nature of each (heterogeneous) agent's local model.

Motivated by this, we propose DiPLe: an algorithm for Directed Personalized Learning. DiPLe enables each agent to learn a personalized model in a fully decentralized setting (i.e., without a central server) through peer-to-peer collaboration with a carefully selected subset of other agents. This is done by iterating through a two-stage process: learning the collaboration graph, followed by distributed learning over the learned graph. Specifically, to learn the collaboration graph, each agent requests local models from other agents, and applies a mixture of each peer model with its own local model to its local dataset. The performance of this mixed model on the local data is used as an assessment of that peer's "usefulness", and the agent unilaterally adjusts its links (including the weights) with other agents based on these assessments. After updating the collaboration graph, agents train their models collaboratively through peer-to-peer learning over this graph.

We show that DiPLe has the following advantages: (1) compared to (centrally coordinated) federated learning algorithms, it achieves personalized learning by selectively communicating with agents who have similar or relevant local datasets, (2) compared to fully connected or arbitrary information exchange graphs, it reduces communication overhead, and (3) outperforms other existing methods for personalized learning, including by allowing for unilateral information exchange between agents.

A distinct feature of DiPLe (including compared to prior work as detailed in Section II) is the *directed* nature of the learned collaboration graph. Intuitively, a directed graph can bring significant benefits to peer-to-peer personalized learning. There are several factors that influence whether an externally trained model is beneficial to the local learning of an agent: the external model's accuracy, the dataset size from which the model is obtained, and the difference in the data distribution

This work has been supported in part by NSF grants CNS-2112471 and CCF-2144283.

between the two agents. As a result, the local models of a pair of agents may have *asymmetric* effects on each other. For example, an agent with a small dataset and low accuracy may place more weight on the model obtained from a peer with a large dataset and a high accuracy model, while the peer will prefer to rely only on its own local model. Our proposed method for learning directed graphs thus enables collaborative learning in settings where the local models of a pair of agents may have different value to each other.

In the remainder of the paper, we first review related work in Section II. We present our algorithm, `DiPLe`, in Section III, and provide analytical results on the generalization error bounds and convergence of our proposed learning method in Section IV. We illustrate the performance of `DiPLe` through numerical experiments, and discuss its advantages compared to a number of existing federated learning and personalized learning algorithms in Section V.

II. Related Work

Decentralized and peer-to-peer learning has been studied in many prior works, (e.g., [2], [3]). The main difference of our approach with this literature is in the desired learning outcome: much of the existing works study peer-to-peer learning algorithms that will lead to consensus on a shared model, while we are interested in collaborative training of personalized models.

Peer-to-peer *personalized* learning has been studied much less extensively; recent work includes [4]–[6]. One key difference of our work with [4] and [5] is that these prior works assume the communication graphs to be given and fixed a priori. In contrast, we study a graph that emerges adaptively as part of the learning process. To the best of our knowledge, the only prior work exploring a similar problem of adaptively learning a collaboration graph is that of Zantedeschi et al. [6]. The algorithm in [6] adjusts the collaboration graph's edges based on the similarity between the local models of a pair of nodes; this leads to an *undirected* collaboration graph. By contrast, we propose a different method for identifying similar agents which ultimately leads to learning a *directed* collaboration graph. We show that learning under such directed collaboration graphs can outperform undirected ones.

Personalized *federated learning* has received more attention in recent work. Smith et al. [7] proposed the MOCHA algorithm for multi-task FL. Fallah et al. [8] proposed a personalized FL algorithm based on the Model-Agnostic Meta-Learning (MAML) framework. The works of [9]–[12] propose a "mixture of experts" approach in which agents maintain personalization by adopting an appropriately weighted sum of the globally trained model with their local model. Clustered FL methods have been proposed in [12]–[15], where agents are assigned to different clusters based on their similarities, and learn collaboratively within these clusters.

Among these works, and setting aside the main difference that unlike these works our method does not need a central server, [10], [12], [16] are most closely related to our work. In these works, similar to our proposed method, agents aim to assess the "usefulness" of either the global model or another agent's model based on its performance on their own local dataset. In particular, the algorithms in [10] and [12] identify a mixing weight between the *global* and local models; by contrast, we do not train a global model, and allow agents to co-train their models with a subset of other agents. In Zhang et al. [16]'s algorithm, each agent identifies all neighbors' models that outperform its own model on the local dataset, and then uses this information together with a first-order Taylor series approximation to build an appropriately weighted mixture model of these neighbors' models. In contrast, we identify the optimal mixing weights by adjusting the weights of a mixed model using a bisection method. Additionally, while [16] updates the mixture weights after each round of training, we freeze the collaboration graph in stage II of `DiPLe` and conduct peer-to-peer learning over it.

III. Problem Formulation and Algorithm

A. Notation and problem formulation

We consider a set of N agents/clients. Let D_i be the local dataset of agent $i \in \{1, \dots, N\}$. Each dataset consists of datapoints $\xi = (\mathbf{x}, y)$ with features $\mathbf{x} \in \mathcal{X}$ and labels $y \in \mathcal{Y}$. The prediction models or hypothesis are denoted by $h \in \mathcal{H}$, where $\mathcal{H}$ is the hypotheses class. Given a loss function $l : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$, the goal of agent i is to find a hypothesis $h \in \mathcal{H}$ that minimizes the expected loss $\mathcal{L}_{D_i}(h) := \mathbb{E}_{(\mathbf{x},y)\sim D_i}[l(h(\mathbf{x}), y)]$. Formally, the *local minimizer* of agent i is

$$h_{loc,i} = \arg\min_{h \in \mathcal{H}} \ \mathcal{L}_{D_i}(h) \ . \tag{1}$$

Note that $h_{loc,i}$ minimizes the loss of agent i on its local data distribution, which is in general different across agents. We assume hypothesis h is linear, and can be equivalently represented by a parameter vector $\mathbf{w} \in \mathbb{R}^d$, and use these interchangeably. We also use $f_i(\mathbf{w}) = \mathcal{L}_{D_i}(h)$ to denote the empirical risk of agent i on its local dataset under a model with parameters $\mathbf{w}$.

As noted in the introduction, our algorithm learns a directed and weighted collaboration graph, denoted $A \in \mathbb{R}^{n \times n}$, which determines the information exchange between agents during peer-to-peer learning. We use α_{ij} to denote the i-th row and j-th column of A. These weights are normalized such that $\alpha_{ii} + \sum_j \alpha_{ij} = 1$ for all i.

The edges in this graph determine the flow of information, as well as the weights in the collaboratively learned *mixture models*: each agent i's final learned model will be of the form $\bar{h}_i = \alpha_{ii} h_i + \sum_j \alpha_{ij} h_{loc,j}$, where $h_{loc,j}$ is the local minimizer model communicated by neighbor j, and h_i is the *mixture minimizer* chosen by agent i as follows:

$$h_i = \arg\min_{h \in \mathcal{H}} \ \mathcal{L}_{D_i}(\alpha_{ii} h + \sum_j \alpha_{ij} h_{loc,j}) \tag{2}$$

Our algorithm learns a collaboration graph A such that the collaboratively learned mixture model $\bar{h}_i = \alpha_{ii} h_i + \sum_j \alpha_{ij} h_{loc,j}$ over this graph will outperform the independently learned local model $h_{loc,i}$ in (1), in terms of generalization error and learning loss on agent i's local data distribution.

B. DiPLe: Directed Personalized Learning

Our proposed algorithm is shown in Algorithm 1. It consists of two stages: updating the collaboration graph, followed by peer-to-peer learning over the learned graph. The algorithm repeats these two stages for R rounds, as detailed below.

At the start of the algorithm, a collaboration graph A is initialized to a matrix with all entries set to 0.5, and the local models $\mathbf{w}_{loc,i}$ are initialized arbitrarily. Then, at each round $r \in [R]$, the following two-stage process is performed:

Stage I: learning the directed graph: The following steps are performed by each agent i in parallel:

1) Update the model from $\mathbf{w}_{loc,i}^{r,-1}$ to $\mathbf{w}_{loc,i}^{r,0}$ by each agent i on its local data D_i separately, using E stochastic gradient descent (SGD) updates. The goal of this training is to minimize the local loss at agent i, as shown in (1).
2) Based on the current graph, the obtained models from step 1 are transferred to neighboring agents. Specifically, if agent i has a directed edge with weight $\alpha_{ij}^{r-1} > 0$ to agent j, it requests the local model $\mathbf{w}_{loc,j}^{r,0}$ of agent j.
3) The graph weights α_{ij} are updated using a *bisection method* with threshold ϵ to solve the following problem:

$$\min \quad f_i((1-\alpha_{ij})\mathbf{w}_{loc,i}^{r,0} + \alpha_{ij}\mathbf{w}_{loc,j}^{r,0}), \tag{3}$$

where if $\alpha_{ij} < \frac{\epsilon}{2}$, it will be set to 0. In words, this is the optimal mixture between agents i and j's local models that would minimize i's local loss.

These steps lead to an updated collaboration graph A^r, which is used in the following stage.

Stage II: Collaborative learning on the directed graph: The agents train their models by repeatedly exchanging updates over collaboration graph A^r for T time steps. At each time $t \in [T]$, the following steps are performed by each agent i in parallel:

1) Update the local minimizer $\mathbf{w}_{loc,i}^{r,t}$ to $\mathbf{w}_{loc,i}^{r,t+1}$ using one SGD update to minimize loss on the local dataset D_i.
2) Request the local models $\mathbf{w}_{loc,j}^{r,t+1}$ from neighbors j with whom the graph edge weight $\alpha_{ij}^r > 0$.
3) The mixture model $\bar{\mathbf{w}}_i^{r,t+1} = \alpha_{ii}^r \mathbf{w}_i^{r,t+1} + \sum_j \alpha_{ij}^r \mathbf{w}_{loc,j}^{r,t+1}$ of agent i is updated using one SGD update on the mixture minimizer model $\mathbf{w}_i^{r,t}$ to minimize the mixture model's loss on the local data (i.e., with objective function (2)):

$$\mathbf{w}_i^{r,t+1} = \mathbf{w}_i^{r,t} - \eta_t \nabla_{\mathbf{w}_i} f_i(\alpha_{ii}^r \mathbf{w}_i^{r,t} + \sum_j \alpha_{ij}^r \mathbf{w}_{loc,j}^{r,t}; \xi_i^{r,t}), \tag{4}$$

where η_t is the SGD step size, and $\xi_i^{r,t}$ is the uniformly randomly sampled data of agent i at round r iteration t.

At the end of stage II, we select the best performing model among $\{\mathbf{w}_{loc,i}^{r,t}, \bar{\mathbf{w}}_i^{r,t}\}_{t=1}^T$ (all local and mixture models trained by agent i during stage II), to initialize the agent's local minimizer model $\mathbf{w}_{loc,i}^{r+1,-1}$ for the next round. Intuitively, such update can help agent i by providing valuable model information from relevant neighbors, and even help this agent escape local optima.

Algorithm 1: DiPLe

Input: Threshold ϵ for bisection method; step sizes $\{\eta_t\}$ for SGD; max. training epochs R, E, T
Output: Personalized models $\mathbf{w}_i$ for $i \in [N]$
Initialize: Local models $\mathbf{w}_{loc,i}^{1,0}$ for $i \in [N]$; weight matrix $A^0 = [0.5]^{N,N}$ of collab. graph
for $r = 1, ..., R$ **do**
  Agents $i \in [N]$ **in parallel do**
    - E steps SGD to update local model parameters from $\mathbf{w}_{loc,i}^{r,-1}$ to $\mathbf{w}_{loc,i}^{r,0}$
    - Request local models $\mathbf{w}_{loc,j}^{r,0}$ from $j \in [N]$ with $\alpha_{ij}^{r-1} > 0$
    - Update α_{ij}^r by *bisection method* with threshold ϵ to minimize the empirical risk $f_i((1-\alpha_{ij}^r)\mathbf{w}_{loc,i}^{r,0} + \alpha_{ij}^r \mathbf{w}_{loc,j}^{r,0})$ of agent i
    - Normalize weights s.t. $\alpha_{ii}^r + \sum_j \alpha_{ij}^r = 1$
  for $t = 0, ..., T-1$ **do**
    Agents $i \in [N]$ **in parallel do**
      - One step SGD to update local model parameters from $\mathbf{w}_{loc,i}^{r,t}$ to $\mathbf{w}_{loc,i}^{r,t+1}$
      - Request local models $\mathbf{w}_{loc,j}^{r,t+1}$ from $j \in [N]$ with $\alpha_{ij}^r > 0$
      - Update mixture minimizer $\mathbf{w}_i^{r,t+1} = \mathbf{w}_i^{r,t} - \eta_t \nabla_{\mathbf{w}_i} f_i(\alpha_{ii}^r \mathbf{w}_i^{r,t} + \sum_j \alpha_{ij}^r \mathbf{w}_{loc,j}^{r,t}; \xi_i^{r,t})$
      - Obtain mixture model $\bar{\mathbf{w}}_i^{r,t+1} = \alpha_{ii}^r \mathbf{w}_i^{r,t+1} + \sum_j \alpha_{ij}^r \mathbf{w}_{loc,j}^{r,t+1}$
  end
  All agents $i \in [N]$**:**
  $\mathbf{w}_{loc,i}^{r+1,-1} = \arg\min_{\mathbf{w} \in \{\mathbf{w}_{loc,i}^{r,t}, \bar{\mathbf{w}}_i^{r,t}\}_{t=1}^T} f_i(\mathbf{w})$
end
Return $\mathbf{w}_i = \mathbf{w}_{loc,i}^{R+1,-1}$, for $i \in [N]$

IV. Analytical Results

In this section, we show two analytical results for our proposed algorithm: the convergence of the learned mixture models, and bounds on its generalization error. For both of these, we assume a fixed directed collaboration graph, i.e., we show these properties for the second stage of DiPLe. We also discuss how the generalization error is affected by the "quality" of the learned collaboration graph.

A. Convergence of the learned mixture models

We begin by showing the convergence of the learned mixture models in stage II of DiPLe over a given graph A. We make the following assumptions on these functions.

Assumption 1: The variance of SGD is bounded: $\mathbb{E}[||\nabla f_i(\mathbf{w}_{loc,i}^t) - \nabla f_i(\mathbf{w}_{loc,i}^t; \xi_i^t)||^2] \leq \sigma^2, i \in V, t \in [T]$.

Assumption 2: $f_i(\cdot)$ are μ-strongly convex: $f_i(\mathbf{w}) \leq f_i(\mathbf{v}) + (\mathbf{w}-\mathbf{v})^T \nabla f_i(\mathbf{v}) + \frac{\mu}{2}||\mathbf{w}-\mathbf{v}||^2$, for all $\mathbf{w}, \mathbf{v} \in \mathbb{R}^n$, $i \in V$.

Assumption 3: $f_i(\cdot)$ are L-Lipschitz continuous: $|f_i(\mathbf{w}) - f_i(\mathbf{v})| \leq L|\mathbf{w}-\mathbf{v}|$, for all $i \in V$.

Under these assumptions, and with appropriate choice of step sizes in SGD, we obtain the following convergence result.

Theorem 1: Assume Assumptions 1 to 3 hold. Let the learning rates be such that $\eta_1 \leq 1$ and $\eta_t = \frac{\beta}{t+r}$, where $r > 0$ and $\beta \leq \min\{1+r, \frac{1+r}{(\alpha_{ii}^2 - \frac{K}{n})\mu}\}$, with $K := |\{j \in [N] \backslash \{i\} |\ \alpha_{ij} > 0\}|$, and n is chosen to ensure $0 < (\alpha_{ii}^2 - \frac{K}{n})\mu\eta_t < 1$. Then, $\bar{\mathbf{w}}_i^t$ in stage II of DiPLe will convergence to $\bar{\mathbf{w}}_i$, the optimal mixed model of agent i, with a rate of $O(\frac{1}{t})$.

A sketch of the proof is given in the Appendix. The full proof can be found in the supplementary material [17].

B. Generalization error bound of the learned models

We also provide a bound on the generalization error of the mixed models on each agent's local data. We present this for the squared hinge loss in classification tasks, which is $l(h(\mathbf{x}), y) = (\max\{0, 1 - yh(\mathbf{x})\})^2$. In the following, we define the local *empirical risk* minimizer of agent i by

$$\hat{h}_{loc,i} = \arg\min_{h \in \mathcal{H}} \quad \hat{\mathcal{L}}_{D_i}(h) \ . \tag{5}$$

The mixture *empirical risk* minimizers are defined similarly

$$\hat{h}_i = \arg\min_{h \in \mathcal{H}} \quad \hat{\mathcal{L}}_{D_i}(\alpha_{ii} h + \sum_j \alpha_{ij} \hat{h}_{loc,j}) \ . \tag{6}$$

We also use the following definition (from [10]).

Definition 1: Let $\mathcal{S}$ be a fixed set of samples. The worst-case disagreement between two pairs of models is defined as

$$\lambda_{\mathcal{H}}(S) = \sup_{h,h' \in \mathcal{H}} \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x},y) \in \mathcal{S}} |h(\mathbf{x}) - h'(\mathbf{x})| \ .$$

The generalization bound on DiPLe's models is as follows.

Theorem 2: Let $\mathcal{H}$ be a hypothesis class with a VC dimension d. Assume the loss function l is Lipschitz continuous with a constant L, and bounded in $[0, P]$. Then, with probability at least $1 - \delta$ there exists a constant C such that the risk of the mixed model $\alpha_{ii}\hat{h}_i + \sum_j \alpha_{ij}\hat{h}_{loc,j}$ on the local distribution D_i is bounded by

$$\begin{aligned}
&\mathcal{L}_{D_i}(\alpha_{ii}\hat{h}_i + \sum_j \alpha_{ij}\hat{h}_{loc,j}) \\
&\leq N\alpha_{ii}^2[\mathcal{L}_{D_i}(h_{loc,i}) + 2C\sqrt{\tfrac{d+\log(1/\delta)}{m_i}} + L\lambda_{\mathcal{H}}(\mathcal{S}_i)] \\
&+ N\sum_j \alpha_{ij}^2[\hat{\mathcal{L}}_{D_j}(\hat{h}_{loc,j}) + C\sqrt{\tfrac{d+\log(1/\delta)}{m_j}} + P||D_i - D_j||_1]
\end{aligned} \tag{7}$$

where S_i is the training data drawn from D_i and m_i is its size, and $||D_i - D_j||_1 = \int |P_{(\mathbf{x},y)\sim D_i} - P_{(\mathbf{x},y)\sim D_j}| \mathrm{d}\mathbf{x}\mathrm{d}y$.

The proof is provided in the supplementary material [17].

Intuition and the impact of the quality of the collaboration graph. Intuitively, the bound indicates that if D_i and D_j are similar, then mixing with neighbor j will not increase the risk at agent i by much. Therefore, the mixing weight α_{ij} can be made larger. Similarly, if the local model of agent j is such that it has high local empirical risk $\hat{\mathcal{L}}_{D_j}(\hat{h}_{loc,j})$ itself, then a large mixing weight with this neighbor will increase the risk at agent i, too. Lastly, it is beneficial to mix with neighbors who have larger local datasets m_j. Our algorithm for learning the collaboration graph adjusts the weights following similar logic: it tries to increase the edge weights with peers that have similar, as well as well-performing, local models.

V. Numerical Experiments

A. Experimental setup

Datasets and prediction models. We evaluate DiPLe and compare it against other algorithms using experiments on two datasets: EMNIST [18] (which consists of 62 classes of 28×28 grayscale images of digits or letters) and CIFAR-10 [19] (which consists of 10 classes of 32×32 color images, including animals and vehicles). We trained a neural network for every agent in our experiments, with models of CIFAR-10 and EMNIST datasets following [20] and [12], respectively.

Local dataset setup. To obtain non-IID local datasets, we follow the same procedure as [21] and select each agent's local data to consist of only two classes, randomly selected from the classes in the corresponding dataset.

Baselines. We compare **DiPLe** against the following:

- **Local training**, where each agent learns a model independently on its local dataset.
- **FedAvg** [21], in which agents collaborate, with the aid of a central server, to learn a shared global model.
- **FedAvg+non-iid** [22], a variant of FedAvg designed to improve over it when local datasets are non-IID.
- **APFL** [10] and **FedAvg+Mapper** [12], two personalized learning algorithms which learn an optimal mixture between a *global* model and each agent's local model.

For each of these, we perform a hyperparameter sweep and choose the one that performs best on the validation set to compare against (additional details are given in [17]). For DiPLe, we choose E = 5, T = 5, R = 20, and $\epsilon = 0.05$.

B. Experimental results and discussion

Table I compares DiPLe's learning accuracy on the local dataset against four baselines, on two datasets, and for $N = 15$ and $N = 100$ agents. In the case of 15 agents, stage I of DiPLe learns a directed graph on all agents. In the case of 100 agents, each agent in DiPLe randomly samples 20 agents to learn a new collaboration graph for that round. From the results in Table I, we can see that our algorithm outperforms all baselines on both datasets and agent sizes. It is also interesting to note that DiPLe surpasses local training more when $N = 100$ in EMNIST, and in CIFAR-10 compared to EMNIST; this is because in these settings, given our method of generating local non-IID datasets and the number of classes in each dataset, it is more likely that agents will encounter a peer with similar local data to collaborate with.

Figure 1 further illustrates the convergence speed of DiPLe (vs. FedAvg and APFL) in an experiment on the CIFAR-10 dataset with 8 agents. This figure implies that DiPLe can identify "useful" peers early on, and refine the performance of the learned model through additional training afterwards.

Figure 2 and Table II compare the performance of DiPLe (which learns over directed graphs) and learning over a similarity-based undirected graph. The four agents in this experiment are divided into two groups, and the datasets for the two agents in the same group are IID (with 2 classes selected randomly from CIFAR-10), but the datasets between

TABLE I
COMPARISON OF LOCAL ACCURACY (PERSONALIZATION)

Algorithm	EMNIST		CIFAR-10	
	15 agents	100 agents	15 agents	100 agents
Local Training	99.70%	94.99%	87.20%	82.09%
FedAvg	90.84%	70.51%	62.76%	54.98%
FedAvg+non-iid	81.33%	71.23%	56.13%	52.72%
FedAvg+Mapper	98.04%	96.62%	65.47%	65.13%
APFL	97.22%	96.30%	78.01%	76.69%
DiPLe	**99.76%**	**98.53%**	**92.33%**	**84.88%**

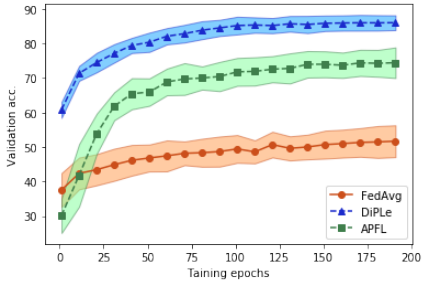

Fig. 1. Validation accuracy vs. training epochs (8 agents' avg. over 10 runs)

the two groups are non-IID (the 2 sampled classes are in general different between groups). Learning results on the undirected graph are based on running FedAvg between the pair of agents with IID local data (this is akin to clustered FL). We observe that while DiPLe learns more slowly (due to the time spent learning the best collaboration graph), the average accuracy obtained by DiPLe is eventually higher than that learned on the undirected graph. We also observe that agents 2 and 3 have benefited from being connected with agents 0 and 1 (and 0 from connecting to 2 and 3), even though their datasets are not similar. Finally, we note that agents are grouped in the undirected graph assuming their datasets are known to be similar. In contrast, DiPLe does not need to know the data distribution in advance.

VI. CONCLUSION

To enable collaborative learning of personalized models when local datasets are heterogeneous, we proposed DiPLe: a fully decentralized peer-to-peer learning approach that trains personalized local models by collaborating only with "similar" agents. We provided analytical characterizations for our learned models' convergence and generalization error. Experimental results on EMNIST and CIFAR-10 datasets highlighted the performance of DiPLe. In particular, the experiments show that DiPLe guides agents to decrease their collaboration with some agents while increasing their interactions with others, thus avoiding negative interference from models of agents with different data/tasks, and outperforming federated learning, independent local training, and other existing personalized learning methods, in learning personalized models.

The main directions of extending our work include safeguarding DiPLe against potential adversarial/privacy attacks, and reducing its communication costs. In particular, our (distributed) algorithm inevitably incurs higher communication costs than (centralized) FL algorithms (e.g, FedAvg, APFL) by removing the central server. Nonetheless, the main purpose of our work is to achieve personalized learning, which as we show, is not achievable by these lower communication algorithms. That said, further reducing DiPLe's communication costs would be of interest to make it more scalable.

TABLE II
LOCAL ACCURACY (PERSONALIZATION), CIFAR-10 NON-IID DATASETS

Algorithm	Agent Number				Average
	0	1	2	3	
Undirected	85.42%	**88.33%**	87.50 %	87.58%	87.21%
DiPLe	**86.87%**	86.13%	**88.53%**	**87.60%**	**87.28%**

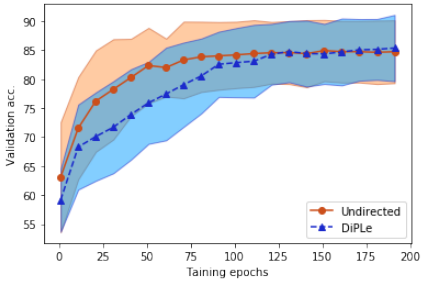

Fig. 2. Validation accuracy vs. training epochs (4 agents' avg. over 10 runs)

APPENDIX

We present a proof sketch for Theorem 1. Our proof is similar to those of [10] and [22]. We first show that the mixture model of i is updated as follows:

$$\bar{\mathbf{w}}_i^{t+1} = \bar{\mathbf{w}}_i^t - \eta_t \alpha_{ii}^2 \nabla f_i(\bar{\mathbf{w}}_i^t; \xi_i^t) - \eta_t \sum_j \alpha_{ij} \nabla f_j(\mathbf{w}_{loc,j}^t; \xi_j^t)$$

Using this, we can bound the reduction in the distance of each update with the optimal mixture model $\bar{\mathbf{w}}_i$ as follows:

$$\mathbb{E}[||\bar{\mathbf{w}}_i^{t+1} - \bar{\mathbf{w}}_i||^2] \leq \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] + {\eta_t}^2 \sigma^2 (\alpha_{ii}^2 + 1 - \alpha_{ii})$$
$$+ \eta_t^2 \mathbb{E}[||\alpha_{ii}^2 \nabla f_i(\bar{\mathbf{w}}_i^t) + \sum_j \alpha_{ij} \nabla f_j(\mathbf{w}_{loc,j}^t)||^2]$$
$$-2\eta_t {\alpha_{ii}}^2 \mathbb{E}[\langle \nabla f_i(\bar{\mathbf{w}}_i^t), \bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i \rangle]$$
$$-2\eta_t \sum_j \alpha_{ij} \mathbb{E}[\langle \nabla f_j(\mathbf{w}_{loc,j}^t), \bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i \rangle]$$

The remainder of the proof proceeds by upperbounding each of the three expected values on the right-hand side of the above inequality (these bounds require the Lipschitz continuity and μ-strong convexity of f_i). This leads to

$$\mathbb{E}[||\bar{\mathbf{w}}_i^{t+1} - \bar{\mathbf{w}}_i||^2]$$
$$\leq (1 - (\alpha_{ii}^2 - \frac{K}{n})\mu\eta_t)\mathbb{E}[||\bar{\mathbf{w}}_i^{t+1} - \bar{\mathbf{w}}_i||^2] + B\eta_t^2$$

where B is a constant that depends on the graph edge weights, among other things. Lastly, we use the above together with the conditions on the stepsizes η_t to show that $\mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] \leq \frac{v}{t+r}$ through induction, completing the proof.

REFERENCES

[1] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[2] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *arXiv preprint arXiv:1705.09056*, 2017.

[3] M. R. Behera, S. Upadhyay, S. Shetty, and R. den Otter, "Federated learning using peer-to-peer network for decentralized orchestration of model weights," 2021.

[4] I. Almeida and J. Xavier, "Djam: Distributed jacobi asynchronous method for learning personal models," *IEEE Signal Processing Letters*, vol. 25, no. 9, pp. 1389–1392, 2018.

[5] P. Vanhaesebrouck, A. Bellet, and M. Tommasi, "Decentralized collaborative learning of personalized models over networks," in *Artificial Intelligence and Statistics*, pp. 509–517, PMLR, 2017.

[6] V. Zantedeschi, A. Bellet, and M. Tommasi, "Fully decentralized joint learning of personalized models and collaboration graphs," in *International Conference on Artificial Intelligence and Statistics*, pp. 864–874, PMLR, 2020.

[7] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," *arXiv preprint arXiv:1705.10467*, 2017.

[8] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," *arXiv preprint arXiv:2002.07948*, 2020.

[9] E. L. Zec, O. Mogren, J. Martinsson, L. R. Sütfeld, and D. Gillblad, "Federated learning using a mixture of experts," *arXiv preprint arXiv:2010.02056*, 2020.

[10] Y. Deng, M. M. Kamani, and M. Mahdavi, "Adaptive personalized federated learning," *arXiv preprint arXiv:2003.13461*, 2020.

[11] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," *arXiv preprint arXiv:2002.05516*, 2020.

[12] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *arXiv preprint arXiv:2002.10619*, 2020.

[13] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3710–3722, 2020.

[14] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19586–19597, 2020.

[15] M. Nafea, E. Shin, and A. Yener, "Proportional fair clustered federated learning," in *2022 IEEE International Symposium on Information Theory (ISIT)*, pp. 2022–2027, 2022.

[16] M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Alvarez, "Personalized federated learning with first order model optimization," *arXiv preprint arXiv:2012.08565*, 2020.

[17] X. Zheng, P. Naghizadeh, and A. Yener, "Diple: Learning directed collaboration graphs for peer-to-peer personalized learning." https://drive.google.com/file/d/1EtpV0psbY98TYfjA0avEh0HdJvr-M6Vc/view?usp=sharing, 2022.

[18] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *2017 international joint conference on neural networks (IJCNN)*, pp. 2921–2926, IEEE, 2017.

[19] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[20] C. T Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with Moreau envelopes," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21394–21405, 2020.

[21] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017.

[22] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[23] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *International Conference on Machine Learning*, pp. 4615–4625, PMLR, 2019.

A. *Proof of Theorem 1*

Our proof is similar to those of [10] and [22].

We first note that the stochastic gradient of agent i's local empirical loss with respect to its local mixture model is given by

$$\begin{aligned}\nabla_{\mathbf{w}_i} f_i(\bar{\mathbf{w}}_i^t;\xi_i^t) &= \nabla_{\mathbf{w}_i} f_i(\alpha_{ii}\mathbf{w}_i^t + \sum_j \alpha_{ij}\mathbf{w}_{loc,j}^t;\xi_i^t)\\ &= \alpha_{ii}\nabla f_i(\alpha_{ii}\mathbf{w}_i^t + \sum_j \alpha_{ij}\mathbf{w}_{loc,j}^t;\xi_i^t)\\ &= \alpha_{ii}\nabla f_i(\bar{\mathbf{w}}_i^t;\xi_i^t) \end{aligned} \tag{8}$$

Using this, the mixed model of agent i is updated as follows:

$$\begin{aligned}\bar{\mathbf{w}}_i^{t+1} &= \alpha_{ii}\mathbf{w}_i^{t+1} + \sum_j \alpha_{ij}\mathbf{w}_{loc,j}^{t+1}\\ &= \alpha_{ii}(\mathbf{w}_i^t - \eta_t\alpha_{ii}\nabla f_i(\bar{\mathbf{w}}_i^t;\xi_i^t)) + \sum_j \alpha_{ij}(\mathbf{w}_{loc,j}^t - \eta_t\nabla f_j(\mathbf{w}_{loc,j}^t;\xi_j^t))\\ &= \alpha_{ii}\mathbf{w}_i^t + \sum_j \alpha_{ij}\mathbf{w}_{loc,j}^t - \eta_t\alpha_{ii}^2\nabla f_i(\bar{\mathbf{w}}_i^t;\xi_i^t) - \eta_t\sum_j \alpha_{ij}\nabla f_j(\mathbf{w}_{loc,j}^t;\xi_j^t)\\ &= \bar{\mathbf{w}}_i^t - \eta_t\alpha_{ii}^2\nabla f_i(\bar{\mathbf{w}}_i^t;\xi_i^t) - \eta_t\sum_j \alpha_{ij}\nabla f_j(\mathbf{w}_{loc,j}^t;\xi_j^t) \end{aligned} \tag{9}$$

Based on this, we assess the norm 2 distance of the learned mixed model $\bar{\mathbf{w}}_i^{t+1}$ at time $t+1$ from the optimal mixture model $\bar{\mathbf{w}}_i$:

$$\begin{aligned}&\mathbb{E}[||\bar{\mathbf{w}}_i^{t+1} - \bar{\mathbf{w}}_i||^2]\\ &= \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i - \eta_t\alpha_{ii}^2\nabla f_i(\bar{\mathbf{w}}_i^t;\xi_i^t) - \eta_t\sum_j \alpha_{ij}\nabla f_j(\mathbf{w}_{loc,j}^t;\xi_j^t)||^2]\\ &= \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i - \eta_t\alpha_{ii}^2\nabla f_i(\bar{\mathbf{w}}_i^t) - \eta_t\sum_j \alpha_{ij}\nabla f_j(\mathbf{w}_{loc,j}^t)||^2]\\ &+\eta_t^2\mathbb{E}[||\alpha_{ii}^2(\nabla f_i(\bar{\mathbf{w}}_i^t) - \nabla f_i(\bar{\mathbf{w}}_i^t;\xi_i^t)) + \sum_j \alpha_{ij}(\nabla f_j(\mathbf{w}_{loc,j}^t) - \nabla f_j(\mathbf{w}_{loc,j}^t;\xi_j^t))||^2]\\ &\leq \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i - \eta_t\alpha_{ii}^2\nabla f_i(\bar{\mathbf{w}}_i^t) - \eta_t\sum_j \alpha_{ij}\nabla f_j(\mathbf{w}_{loc,j}^t)||^2] + \eta_t^2\sigma^2(\alpha_{ii}^2 + 1 - \alpha_{ii}) \qquad (10)\\ &= \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] + \underbrace{\eta_t^2\,\mathbb{E}[||\alpha_{ii}^2\nabla f_i(\bar{\mathbf{w}}_i^t) + \sum_j \alpha_{ij}\nabla f_j(\mathbf{w}_{loc,j}^t)||^2]}_{T_1}\\ &\underbrace{-2\eta_t{\alpha_{ii}}^2\mathbb{E}[\langle\nabla f_i(\bar{\mathbf{w}}_i^t), \bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i\rangle]}_{T_2}\\ &\underbrace{-2\eta_t\sum_j \alpha_{ij}\mathbb{E}[\langle\nabla f_j(\mathbf{w}_{loc,j}^t), \bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i\rangle]}_{T_3} + {\eta_t}^2\sigma^2(\alpha_{ii}^2 + 1 - \alpha_{ii}) \qquad (11)\end{aligned}$$

Here, inequality (10) uses Assumption 1 (that the variance of stochastic gradient of each agent i is bounded by σ^2).

First, we bound the term T_1:

$$\begin{aligned}
&\mathbb{E}[||\alpha_{ii}^2 \nabla f_i(\bar{\mathbf{w}}_i^t) + \sum_j \alpha_{ij} \nabla f_j(\mathbf{w}_{loc,j}^t)||^2] \\
&\leq 2\alpha_{ii}^4 \mathbb{E}[||\nabla f_i(\bar{\mathbf{w}}_i^t)||^2] + 2\mathbb{E}[||\sum_j \alpha_{ij} \nabla f_j(\mathbf{w}_{loc,j}^t)||^2] &(12)\\
&\leq 2\alpha_{ii}^4 L^2 + 2L^2 \sum_j \alpha_{ij}^2 &(13)\\
&= 2L^2(\alpha_{ii}^4 + \sum_j \alpha_{ij}^2) &(14)
\end{aligned}$$

The inequality (12) uses the Jensen inequality and the inequality (13) comes from Assumption 3 that $f_i(\cdot)$ are L-Lipschitz continuous, and the triangle inequality.

We also bound T_2 using Assumption 2 that $f_i(\cdot)$ is μ-strongly convex:

$$\begin{aligned}
&-2\eta_t {\alpha_{ii}}^2 \mathbb{E}[\langle \nabla f_i(\bar{\mathbf{w}}_i^t), \bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i \rangle] \\
&\leq -2\eta_t {\alpha_{ii}}^2 (\mathbb{E}[f_i(\bar{\mathbf{w}}_i^t)] + \frac{\mu}{2}\mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] - f_i(\bar{\mathbf{w}}_i)) \\
&= -\mu\eta_t \alpha_{ii}^2 \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] - 2\eta_t \alpha_{ii}^2 \mathbb{E}[f_i(\bar{\mathbf{w}}_i^t) - f_i(\bar{\mathbf{w}}_i)] &(15)
\end{aligned}$$

Finally, we bound T_3 using the Cauchy-Schwarz inequality and AM-GM inequality:

$$\begin{aligned}
&-2\eta_t \sum_j \alpha_{ij} \mathbb{E}[\langle \nabla f_j(\mathbf{w}_{loc,j}^t), \bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i \rangle] \\
&\leq \eta_t \sum_j \alpha_{ij} (\frac{n\alpha_{ij}}{\mu} \mathbb{E}[||\nabla f_j(\mathbf{w}_{loc,j}^t)||^2] + \frac{\mu}{n\alpha_{ij}} \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2]) &(16)\\
&\leq \frac{n\sum_j \alpha_{ij}^2}{\mu} \eta_t L + \frac{K\mu\eta_t}{n} \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] &(17)
\end{aligned}$$

The inequality (16) comes form Cauchy-Schwarz inequality and AM-GM inequality. The inequality (17) comes from Assumption 3 that $f_i(\cdot)$ are L-Lipschitz continuous.

Assuming a diminishing step-size $\eta_t = \frac{\beta}{t+r}$, $r > 0$ and $\eta_1 \leq 1$, we plug back (14), (15), and (17) in (11), and get:

$$\begin{aligned}
&\mathbb{E}[||\bar{\mathbf{w}}_i^{t+1} - \bar{\mathbf{w}}_i||^2] \\
&\leq \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] + {\eta_t}^2 (2L^2(\alpha_{ii}^4 + \sum_j \alpha_{ij}^2)) - \mu\eta_t \alpha_{ii}^2 \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] \\
&- 2\eta_t \alpha_{ii}^2 \underbrace{\mathbb{E}[f_i(\bar{\mathbf{w}}_i^t) - f_i(\bar{\mathbf{w}}_i)]}_{\geq 0} + \frac{n\sum_j \alpha_{ij}^2}{\mu} \eta_t L + \frac{K\mu\eta_t}{n} \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] \\
&+ {\eta_t}^2 \sigma^2 (\alpha_{ii}^2 + 1 - \alpha_{ii}) \\
&\leq (1 - \mu\alpha_{ii}^2 \eta_t + \frac{K\mu\eta_t}{n}) \mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2] \\
&+ {\eta_t}^2 \underbrace{(2L^2(\alpha_{ii}^4 + \sum_j \alpha_{ij}^2) + \frac{n\sum_j \alpha_{ij}^2}{\mu} L + \sigma^2(\alpha_{ii}^2 + 1 - \alpha_{ii}))}_{B} \\
&= (1 - (\alpha_{ii}^2 - \frac{K}{n})\mu\eta_t) \mathbb{E}[||\bar{\mathbf{w}}_i^{t+1} - \bar{\mathbf{w}}_i||^2] + B\eta_t^2 &(18)
\end{aligned}$$

We follow the proof of Theorem 1 of [22] in following parts. Denote $\mathbb{E}[||\bar{\mathbf{w}}_i^t - \bar{\mathbf{w}}_i||^2]$ as $\triangle_t$. For some $\beta \leq \min\{1 + r, \frac{1+r}{(\alpha_{ii}^2 - \frac{K}{n})\mu}\}$ and $v = \max\{\frac{\beta^2 B}{(\alpha_{ii}^2 - \frac{K}{n})\beta\mu - 1}, (r+1)\triangle_1\}$, which ensure $\triangle_1 \leq \frac{v}{r+1}$ and $[\frac{\beta^2 B}{(t+r)^2} - \frac{(\alpha_{ii}^2 - \frac{K}{n})\beta\mu - 1}{(t+r)^2} v] \leq 0$, we will prove $\triangle_t \leq \frac{v}{t+r}$ by induction method.

If $\triangle_t < \frac{v}{r+t}$, we can prove $\triangle_{t+1} < \frac{v}{r+(t+1)}$ as follows:

$$\begin{aligned}
\triangle_{t+1} &\leq (1-(\alpha_{ii}^2 - \frac{K}{n})\mu\eta_t))\triangle_t \\
&\leq (1-(\alpha_{ii}^2 - \frac{K}{n})\frac{\mu\beta}{t+r})\frac{v}{t+r} + \frac{\beta^2 B}{(t+r)^2} \\
&= \frac{t+r-1}{(t+r)^2}v + [\frac{\beta^2 B}{(t+r)^2} - \frac{(\alpha_{ii}^2 - \frac{K}{n})\beta\mu - 1}{(t+r)^2}v] \\
&\leq \frac{v}{r+(t+1)}
\end{aligned} \tag{19}$$

because we know $\triangle_1 < \frac{v}{r+1}$ from the definition of v, the $\triangle_t \leq \frac{v}{t+r}$ is proved.

B. Proof of Theorem 2

Our proof is similar to that of [10], adapted to our peer-to-peer learning setting. Starting from the risk $\mathcal{L}_{D_i}(\alpha_{ii}\hat{h}_i + \sum_j \alpha_{ij}\hat{h}_{loc,j})$, we have

$$\begin{aligned}
&\mathcal{L}_{D_i}(\alpha_{ii}\hat{h}_i + \sum_j \alpha_{ij}\hat{h}_{loc,j}) \\
&= \mathbb{E}_{(\mathbf{x},y)\sim D_i}[(\max\{1 - y(\alpha_{ii}\hat{h}_i + \sum_j \alpha_{ij}\hat{h}_{loc,j})\})^2] \\
&= \mathbb{E}_{(\mathbf{x},y)\sim D_i}[(\alpha_{ii}\max\{1 - y\hat{h}_i\} + \sum_j \alpha_{ij}\max\{1 - y\hat{h}_{loc,j}\})^2] \\
&\leq N\alpha_{ii}^2\mathbb{E}_{(\mathbf{x},y)\sim D_i}[(\max\{1 - y\hat{h}_i\})^2] + N\sum_j \alpha_{ij}^2\mathbb{E}_{(\mathbf{x},y)\sim D_i}[(\max\{1 - y\hat{h}_{loc,j}\})^2] \\
&= N\alpha_{ii}^2\mathcal{L}_{D_i}(\hat{h}_i) + N\sum_j \alpha_{ij}^2\mathcal{L}_{D_i}(\hat{h}_{loc,j})
\end{aligned} \tag{20}$$

Next, using the uniform VC dimension error bound over $\mathcal{H}$ [23], we know

$$|\mathcal{L}_{D_i}(h) - \hat{\mathcal{L}}_{D_i}(h)| \leq C\sqrt{\frac{d + \log(1/\delta)}{m_i}}, \quad \forall h \in \mathcal{H} \ . \tag{21}$$

with probability at least $1 - \delta$.

Then, following techniques similar to those in [10], we can get

$$\begin{aligned}
\mathcal{L}_{D_i}(\hat{h}_i) &\leq \mathcal{L}_{D_i}(h_{loc,i}) + 2C\sqrt{\frac{d + \log(1/\delta)}{m_i}} + \hat{\mathcal{L}}_{D_i}(\hat{h}_i) - \hat{\mathcal{L}}_{D_i}(\hat{h}_{loc,i}) \\
&\leq \mathcal{L}_{D_i}(h_{loc,i}) + 2C\sqrt{\frac{d + \log(1/\delta)}{m_i}} + L\lambda_{\mathcal{H}}(\mathcal{S}_i) \ .
\end{aligned} \tag{22}$$

Lastly, from Lemma 1 in [10] we know that

$$\mathcal{L}_D(h) \leq \mathcal{L}_{D'}(h) + P||D - D'||_1 \ . \tag{23}$$

This in turn means that

$$\mathcal{L}_{D_i}(\hat{h}_{loc,j}) \leq \mathcal{L}_{D_j}(\hat{h}_{loc,j}) + P||D_i - D_j||_1 \ . \tag{24}$$

Using (21), we also have

$$\mathcal{L}_{D_j}(\hat{h}_{loc,j}) \leq \hat{\mathcal{L}}_{D_j}(\hat{h}_{loc,j}) + C\sqrt{\frac{d + \log(1/\delta)}{m_j}} \tag{25}$$

Substituting equations (22), (24), and (25), in (20), we get:

$$\mathcal{L}_{D_i}(\alpha_{ii}\hat{h}_i + \sum_j \alpha_{ij}\hat{h}_{loc,j}) \tag{26}$$

$$\leq N\alpha_{ii}^2\Big(\mathcal{L}_{D_i}(h_{loc,i}) + 2C\sqrt{\frac{d+\log(1/\delta)}{m_i}} + L\lambda_{\mathcal{H}}(\mathcal{S}_i)\Big)$$

$$+ N\sum_j \alpha_{ij}^2\Big(\hat{\mathcal{L}}_{D_j}(\hat{h}_{loc,j}) + C\sqrt{\frac{d+\log(1/\delta)}{m_j}} + P||D_i - D_j||_1\Big) \tag{27}$$

This completes the proof.

C. Additional details on the experimental setup

The hyperparameters we chose and the swept parameter sets are as follows.

- Local training: client num epochs=1, client step size=0.01. The step size is chosen from $\{1, 0.3, 0.1, 0.05, 0.03, 0.01\}$.
- FedAvg [21]: client num epochs=1, client step size=0.03. The step size is chosen from $\{1, 0.3, 0.1, 0.05, 0.03, 0.01\}$.
- FedAvg+non-iid [22]: client num epochs=1, client step size=$\eta_t = \frac{\eta_0}{t+1}$, and $\eta_0 = 0.03$. η_0 is chosen from $\{1, 0.3, 0.1, 0.05, 0.03, 0.01\}$.
- FedAvg+Mapper [12]: client num epochs=1, client step size=0.03. Sever num epochs=1, sever step size=0.05. The client and sever step size are chosen from $\{1, 0.3, 0.1, 0.05, 0.03, 0.01\}$.
- APFL [10]: client num epochs=4, client step size=0.01. Sever num epochs=6, sever step size=0.01. The client and sever step size are chosen from $\{1, 0.3, 0.1, 0.05, 0.03, 0.01\}$, $\alpha = 0.25$.
- `DiPLe`: client num epochs=1, client step size=0.03. The step size is chosen from $\{1, 0.3, 0.1, 0.05, 0.03, 0.01\}$, E = 5, T = 5, R = 20.

D. Additional experiments on `DiPLe`

Table III shows the results for $N = 4$ agents in the non-IID setting of the CIFAR-10 dataset. We note that while the accuracy of `DiPLe` is lower than independent local training for agent 0, the average personalized accuracy and the accuracy on the other three agents is higher with `DiPLe` than both federated learning (fully connected graph) and local training.

TABLE III
LOCAL ACCURACY (PERSONALIZATION), CIFAR-10 NON-IID DATASETS

Algorithm	Agent Number				Average
	0	1	2	3	
Fully connected	59.0%	53.0%	63.5%	54.0%	57.38%
Local Training	**90.5%**	76.00%	82.00%	84.00%	83.13%
`DiPLe`	89.50%	**79.00%**	**87.50%**	**86.50%**	**85.63%**

Table IV also shows the results for $N = 4$ agents in the non-IID setting of the CIFAR-10 dataset. In this experiment, the four agents belong to two different groups, and the data selection of the four agents is the same as Table II. Unlike the experiments in Table II, there are a models transfer between these two groups. When two groups communicate with each other, the weights of models passed to each group are the same, which means a symmetric matrix can represent the weights of the models between agents. In contrast, matrix A in `DiPLe` is asymmetric. We chose 0.3 and 0.6 as the weights between groups in this experiment. We found that except that on agent 1 the result of `DiPLe` is slightly worse than the undirected graph, `DiPLe` surpasses the undirected graph on the other three agents and the average.

TABLE IV
LOCAL ACCURACY (PERSONALIZATION), CIFAR-10 NON-IID DATASETS

Algorithm	Agent Number				Average
	0	1	2	3	
Undirected (weight between group = 0.6)	82.50%	**85.45%**	86.90%	78.55%	83.29%
Undirected (weight between group = 0.3)	82.15%	83.40%	86.90%	76.90%	82.34%
`DiPLe`	**84.27%**	82.20%	**88.53%**	**81.60%**	**84.15%**